ESTIMATION OF DRONE LOCATION USING RECEIVED SIGNAL STRENGTH INDICATOR

Varun Kumar Jagini

Thesis Prepared for the Degree of

MASTER OF SCIENCE

UNIVERSITY OF NORTH TEXAS

August 2021

APPROVED:

Kamesh Namuduri, Major Professor
Ifana Mahbub, Committee Member
Hua Sun, Committee Member
Shengli Fu, Chair of the Department of Electrical Engineering
Hanchen Huang, Dean of the College of Engineering
Victor Prybutok, Dean of the Toulouse Graduate School

Jagini, Varun Kumar. *Estimation of Drone Location Using Received Signal Strength Indicator*. Master of Science (Electrical Engineering), August 2021, 43 pp., 1 table, 16 figures, 26 numbered references.

The main objective of this thesis is to propose a UAV (also called as drones) location estimation system based on LoRaWAN using received signal strength indicator in a GPS denied environment. The drones are finding new applications in areas such as surveillance, search, rescue missions, package delivery, and precision agriculture. Nearly all applications require the localization of UAV during flight. Localization is the method of determining a UAVs physical position using a real or virtual coordinate system. This thesis proposes a LoRaWAN-based UAV location method and presents experimental findings from a prototype. The thesis mainly consists of two different sections: one is the distance estimation and the other is the location estimation. First, the distance is estimated based on the mean RSSI values which are recorded at the ground stations using the path loss model. Later using the slant distance estimation technique, the path loss model parameters L and C are estimated whose values are unknown at the beginning. These values completely depend on the environment. Finally, the trilateration system architecture is employed to find the 3-D location of the UAV.

ACKNOWLEDGEMENTS

I would like to express my heartfelt gratitude to Dr. Kamesh Namuduri, my major advisor and mentor, for his unwavering support and guidance. His insights guided me through my academic career at UNT. I was able to push myself to reach goals with the help of Dr. Kamesh Namuduri. Hopefully, I would also make him proud.

Dr. Ifana Mahbub and Dr. Hua Sun have been invaluable in directing me through my studies and providing useful perspectives. I am grateful for the Department of Electrical Engineering for awarding me a Teaching Assistantship.

Finally, I would like to express my gratitude to all of my friends who have helped me through this difficult time in my life. I would like to thank Ram Charan, Vineeth, Nikitha, Rajitha, Bhavya and Adhya for their support and for being a part of my life.

TABLE OF CONTENTS

Page

LIST OF TABLES AND FIGURES

Page

Tables

Figures

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

Due to their autonomy, versatility, and wide range of application domains, unmanned aerial vehicles (UAVs), also known as drones, have been the subject of intense research in recent years. They are finding new applications in areas such as surveillance, search, rescue missions, package delivery, and precision agriculture. Nearly all applications require the localization of UAV during flight. Localization is the method of determining a UAVs physical position using a real or virtual coordinate system. When we cannot measure the location of UAV directly, in such cases the localization becomes complex. In general, the precision of the approximate position information at a given time is used to test device output with localization.

Until recently, the favored approaches used for drone localization were based on Radar and Global positioning system (GPS). The radar approach is useful for detecting large aircrafts, but it fails with UAVs. The concept of radar cannot be used to localize the UAVs flying at lower altitudes because the electromagnetic waves from the radar cannot reach the UAVs that are very small in size. This drawback limits the use of radar for localizing UAVs. GPS is one of the simplest techniques that can be used to locate UAVs. But GPS does not work in a non-line of sight environment and the energy consumption is larger than other techniques [1]. Because of these limitations the GPS method alone is insufficient for location estimation. Received signal strength indication (RSSI) is one of the methods which overcomes the above limitations and helps us in getting the accurate location of UAV in areas where GPS may be denied. RSSI is the technique that measures the power of a signal strength at the receiver. Thus, for getting the accurate RSSI

values there must be a proper communication between the transmitter and receiver for which wireless methods can be employed.

LoRaWAN is a cloud-based medium access control (MAC) layer protocol that acts as a network layer protocol between LPWAN gateways and end-node devices [2]. This thesis proposes a LoRaWAN-based UAV location method and presents experimental findings from a prototype. The two main components of the proposed system are UAV and ground stations (GS), and both will be using LoRaWAN radios. A total of four ground stations are required to estimate the location of UAV in three dimensions. The UAV broadcasts a sequence of messages that will be received by the GSs. Based on these messages the distance between the GS and UAV is calculated.

1.2 Objective and Significance of the Research

The main objective of this research is to propose a UAV location estimation system based on LoRaWAN using RSSI.

First, the distance is estimated based on the mean RSSI values which are recorded at the ground station using the path loss model. Later using the slant distance estimation technique, the path loss model parameters L and C are estimated whose values are unknown at the beginning. These values completely depend on the environment. Finally, the trilateration system architecture is employed to find the 3-d location of the UAV.

1.3 Organization of Thesis

The rest of the thesis has been organized into several chapters.

Chapter 2 gives a clear idea on LoRaWAN, distance estimation and location estimation and presents the related work.

Chapter 3 is about the components used and architecture. Here a detailed explanation of each component used in the system is explained.

Chapter 4 discusses the algorithm involved in the experiment.

Chapter 5 details about the experimental implementation and the results obtained.

Chapter 6 provides a conclusion and future applications of the thesis.

CHAPTER 2

LITERATURE REVIEW AND RELATED WORK

2.1 LoRaWAN

LoRaWAN is a low power wide area network (LPWAN) which is intended for wireless battery-operated devices. LoRaWAN targets key requirements of the Internet of things such as secure bi-directional communication, mobility, and localization services [3]. LoRaWAN is a one-to-many networking protocol and is one of the wireless technologies that sends the data to longer distances at a very low data rate by using the concept of chirp spread spectrum modulation. An advantage of the chirp spread spectrum modulation method is that the timing and frequency offsets between transmitter and receiver are equivalent, which greatly reduces complexity of the receiver design [4]. This modulation technique and low data rate of the LoRaWAN module led to very low receiver sensitivity. With the help of these LoRaWAN modules we can send data directly to any base station that is LoRaWAN compatible. There are three classes (A, B, C) that support the LoRaWAN technology.

Class A is also called as pure ALOHA system because it is purely asynchronous. This means that the end nodes in the architecture do not wait for any time to communicate with the gateway. The nodes simply transmit the information whenever they need to and lie dormant until then. As soon as one node completes its transmission, another starts immediately.

Class B allows to send the messages to the destination using the gateway. The gateway transmits a message for every 128 seconds. For time synchronization, the transmitter module used in the architecture sends a message at the start of every second.

In class C the downlink message can be sent at any time and allows nodes to listen constantly. To listen constantly a node must be completely active all the time, which leads to lots

of energy consumption.

2.1.1 Seeeduino LoRaWAN

Seeeduino and Moteino LoRaWAN modules are the two modules that implement the LoRaWAN protocol. In this thesis, we will be using Seeeduino LoRaWAN as the communication module for both the UAV and GSs. The main reason for using the Seeeduino module over Moteino is that the Seeeduino LoRaWAN module comes with a built-in wire antenna, while the Moteino LoRaWAN module needs a separate directional antenna for transmitting and receiving the data.

Seeeduino LoRaWAN is an Arduino development board with LoRaWAN protocol embedded in it and works based on the communication module RHF76-052AM. However, Seeeduino LoRaWAN is compatible with only the classes A and C [5]. With the help of the lithium battery management chip, which is embedded on the board, the Seeeduino LoRaWAN module can be charged using the USB interface. A fully charged lithium battery will power the board for several months in low consumption mode. The RSSI values can be recorded by the Seeeduino module, which operates on the 433/868 MHz frequency band.

2.1.2 Features [6]

- Minimum current (3.7V LiPo battery) – 2mA
- ATSAMD21G18 @ 48MHz with 3.3V logic/power
- Arduino compatible (based on Arduino Zero bootloader)
- Embedded with lithium battery management chip and status indicator led
- 20 GPIOs
- 4 on-board Grove connectors
- 18 x PWM pins

- 6 x analog inputs
- 1 x analog output (A0)
- 3.3V regulator with 200mA output
- Reset button.

2.2 Distance Estimation Methods

The following techniques [7] are used to estimate distance between any two nodes.

2.2.1 Time of Arrival (TOA)

TOA is one of the simplest and commonly used ranging techniques. TOA is based on knowing the exact time that a signal was sent from the target, the exact time the signal arrives at a reference point, and the speed at which the signal travels (usually the speed of light) [8]. This is achieved by calculating the time of propagation of the transmitted signal. This TOA method is mainly divided into two types of ranging: one-way ranging (OWR) and two-way ranging (TWR). The time difference present between the transmitted time and received time is calculated in the OWR. Whereas, in TWR the round- trip time (RTT) of the signal propagated is estimated without time synchronization. Time synchronization is the essential component that is required for this technique to be used effectively. TOA requires a highly accurate clock that is synchronized between the nodes if a one-way function is used. By sending any message that contains Time of Departure (ToD) to another node, the other node can determine its distance from the sender, provided the two nodes have synchronized clocks.

The disadvantage of using the TOA method is that it is affected by Non-Line of Sight (NLOS) situations. Also, energy consumption to transmit a signal by using this TOA is very high and computation is very costly.

2.2.2 Angle of Arrival (AOA)

The AOA method estimates the distance of the target by calculating the angle present between the incident wave and a referred direction. The AOA technique generally requires two nodes other than the transmitting node if they are not on a straight line. If these nodes tend to lie in a straight line, in such case an extra node is required to accurately find the location. The AOA method uses the concept of antenna rotation either mechanically or electronically.

The main issue with the AOA method is that it requires extra hardware which increases the cost of hardware and its weight. Also, its accuracy is completely dependent on the directivity of the antennas used which may vary by shadowing and multipath fading. Thus, AOA is not an adequate method for estimating the distance.

2.2.3 Time Difference of Arrival (TDOA)

TDOA is completely based on the difference of the various arrival times of a signal that is transmitted from the source to several destination nodes. At a particular instant of time, data of the signal received at various nodes is used for its implementation. The cross-correlation algorithm is performed between the two signals received at two nodes to estimate the TDOA which yields two hyperbolic curve equations. By using these hyperbolic equations, we can find the distance of the target node. This TDOA is so simple and has many advantages compared with the other techniques. But additional hardware is required for sending two signals and theoretical computation of TDOA is complex.

2.2.4 Received Signal Strength Indication (RSSI)

RSSI is a technique which measures the power of received signal strength at the receiver. The value of RSSI is in decibel (dB) and is generally a negative value for most of the wireless datalinks. Multipath interference affects the values of received signals because of diffraction,

reflection, and scattering. Because of this multipath interference it leads to signal fluctuations in a free space propagation model. Therefore, to improve the accuracy of RSSI the propagation loss is calculated. Using the mathematical formula for RSSI, measured signal value can be equated to the value of distance between the transmitter and the receiver. When there is a change of distance between the transmitter and the receiver, the signal strength varies accordingly. Hence this signal strength can be calculated using the Friis free space equation. Almost all the receivers which are used in any application can be used to estimate the RSSI values. Hence, this technique is very easy to use. As a result, no additional hardware is needed. The cost of computation for estimating the distance using this method is also very low.

2.3 Location Estimation

To estimate the location of any node it requires knowing the information of two things. First, the location of at least three ground stations (for 2D) or four ground stations (for 3D) coordinates and second, the distance between the GS and the target node must be known. For example, there are approximately 31 satellites in the GPS system, where each satellite broadcasts its position and time to the rest of the world. Knowing how far a client is from a satellite, the location of the client can be determined in a spherical orbit.

2.4 Related Work

Most prior works in UAV location estimation use GPS information to estimate the distance and location. The main goal of this research is to propose an alternative technology that can be used for specific applications. From different points of views, there are few works concentrated on studying the RSSI-based location estimation.

For example, Kumar et al. adopted an RSSI-based location estimation technique to estimate the internode distances which is further used for estimating the node's location. They concluded

that the distance-estimated error for RSSI-based location schemes in wireless sensor networks is roughly identical under ideal deployment conditions [9].

Wang et al. designed the GuideLoc system that helps rescue people from a natural disaster using the UAV. This system uses RSSI and the angle of arrival (AoA) of the trapped person to find the location of that person. The system changes the direction based on the strength of the signal. Once the UAV gets over that person, it uses GPS data of that location as the trapped person's location. Our system differs from this system by relying on the RSSI value to estimate the UAV location and not the node's location [10].

Cheng et al. used the RSSI values to solve the issue of locating the UAV in non-line of sight environments. They used the concept of maximum joint probability algorithm to correctly detect the UAV location [11].

Palazon et al. utilized the RSSI-based location scheme to study the location accuracy and how it can be affected when deploying different numbers of anchor nodes [12].

To calculate the location error in an indoor environment, Hamdoun et al. proposed the RSSI-based location algorithm by using multiple antennas at both the transmitter and receiver side in an indoor environment. They found that the location performance for multiple antennas at both sides is better than that for antennas at either the transmitter or the receiver side [13] .

To locate a UAV in an indoor environment, Tian et al. introduce the HiQuadLoc system that uses Wi-Fi access points. There are two phases: online and offline that are used by the system. To detect the location of UAV in the online phase, the offline phase divides the indoor area into cubes with the known RSSI values [14].

Based on RSSI measurements, Awad et al. proposed a distance-based location technique in wireless sensor networks and found that the main effect on the distance measurements is the

power transmission [15].

Our research also uses RSSI values for distance estimation. We use LoRaWAN for location estimation over long distances in the outdoor environment.

CHAPTER 3

SYSTEM COMPONENTS AND ARCHITECTURE

3.1 System Components

There are four main components that are present in our system prototype. These include LoRaWAN modules, GS's, a battery, and a UAV. These components are briefly discussed as follows:

- *LoRaWAN module*: There are a total of 5 LoRaWAN modules used in our prototype. Out of which 4 are used for the ground stations as receivers and the other one is fixed to UAV as transmitter. Reason for using the LoRaWAN module over the Moteino is that it has an onboard antenna which does not require any external hardware. The communication between the transmitter and receiver is established using RHF76-052AM.

- *Ground station:* There are a total of four ground stations used for estimating the drone location. For each ground station we use a regular laptop connected with a LoRaWAN module. Using the Arduino IDE, the laptop is used to configure the LoRaWAN module as receiver. On the serial monitor, the RSSI values obtained are registered. Also, with the help of onboard GPS on the LoRaWAN board, the location of each ground station is obtained and recorded.

- *Antenna*: The Seeeduino LoRaWAN module includes a wire antenna. Communication between the transmitter and various receivers present in ground stations is achieved with the help of these in-built wire antennas.

- *Battery*: The Seeeduino LoRaWAN board has an embedded and integrated lithium battery management chip that allows the board to be charged by USB interface. In the low consumption mode, a fully charged lithium battery (3.7V) can power the board for several months [16] .

- *UAV*: A LoRaWAN module programmed to work as a transmitter is attached to the UAV along with the battery.

The architecture of our system is divided into two main stages: distance and location estimation.

3.2 Architecture for Distance Estimation

To estimate the distance between two nodes such as the UAV and the GSs, one of the methods discussed earlier should be used. In this work, the concept of RSSI is used to estimate the distance. The architecture for distance estimation is as follows:

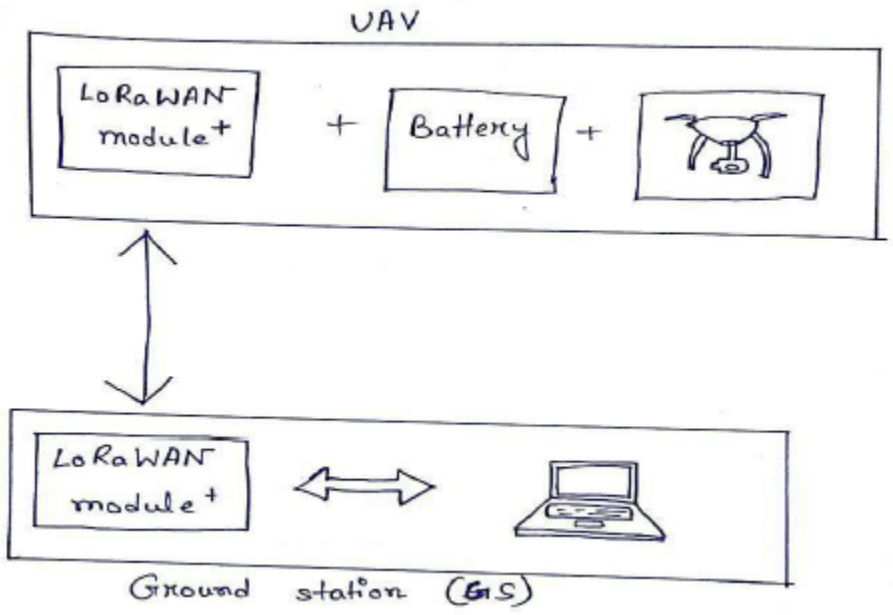

Figure 3.1: Architecture used for estimating the distance.

There are two LoRaWAN modules used for distance estimation: one is fixed to the UAV and powered by battery and the other is connected to a laptop to control and record the data and serves as the GS.

For distance estimation, an unmanned aerial vehicle persistently transmits the data which contains its identification. The LoRaWAN modules maintain a minimum of two seconds interval

time between successive messages to avoid loss of data. The log distance path loss model expresses the following:

$$\mathrm{RSSI} = -10 * \mathrm{L} * \log_{10}(\mathrm{d}) - \mathrm{C} \quad \text{(Eq. 3.1)}$$

in which 'RSSI' refers to Received signal strength indication estimated at the objective, ’d’ refers to distance, 'L' refers to loss exponent path and 'C' refers to a constant. From Eq. 3.1 the distance ‘d’, between UAV and GS is measured as follows:

$$\mathrm{d} = 10^{-((\mathrm{RSSI}-\mathrm{C})/10\mathrm{L})} \quad \text{(Eq. 3.2)}$$

A single value of RSSI is not sufficient to estimate the distance because these values of RSSI are not stable and accurate. Typically, multiple values of RSSI must be used to get an accurate value of the distance. In this experiment, an average of ten RSSI values are used to estimate the distance accurately. Ten is just chosen as a random constant for a proper tradeoff between the time and fluctuation in the RSSI values. Thus, the modified form of Eq. 3.2 is depicted as follows:

$$\mathrm{d} = 10^{-((\mathrm{meanRSSI}-\mathrm{C})/10\mathrm{L})} \quad \text{(Eq. 3.3)}$$

Here the mean RSSI is the average of ten RSSI values. Even though 'C 'and 'L' are persistent in Eq. 3.1, their qualities are at first obscure and rely upon the surrounding. To analyze these boundaries, we require average RSSI esteems and their relating spans for a couple of familiar locations. Hence, a configuration is expected to analyze these qualities utilizing Eq. 3.1. To do so, we fit a linear model to the mean RSSI values. In the resulting model, the slope is -10*L (in this way, L= - inclination/10), and it may be determined as below:

$$\mathrm{slope} = \left(\frac{\sum \mathrm{xy} - \mathrm{n}\overline{\mathrm{xy}}}{\sum x^2 - \mathrm{nx}^{-2}}\right) \quad \text{(Eq. 3.4)}$$

in which, x is average RSSI value at a familiar place or a familiar span, 'y' is the log10(d) value

relating to the average RSSI esteem, 'n' is the quantity of RSSI esteems incorporated for that average, $\overline{x}$ and $\overline{y}$ are the average over all average RSSI esteems and the average over all log10(d) values, respectively. In the subsequent direct configuration, the convergence point is 'C' (subsequently, C= -convergence) that might be determined from a straight configuration as below [16]:

$$\text{intersection} = \overline{y} - \text{slope} \times \overline{x} \quad \text{(Eq. 3.5)}$$

We can use a laser meter to determine the distance between unmanned aerial vehicles as well as a ground station. It gets hard to find such estimations when the real span becomes over 200m. In that case, unmanned aerial vehicles become more tiny and tough to identify with the help of the laser meter.

Hence, as shown in Figure 3.2, we measure the ground distance (GD) between a ground point (GP) and the GS to compute the slant distance (SD) between the GS and the UAV, which equals d in Eq. 3.3.

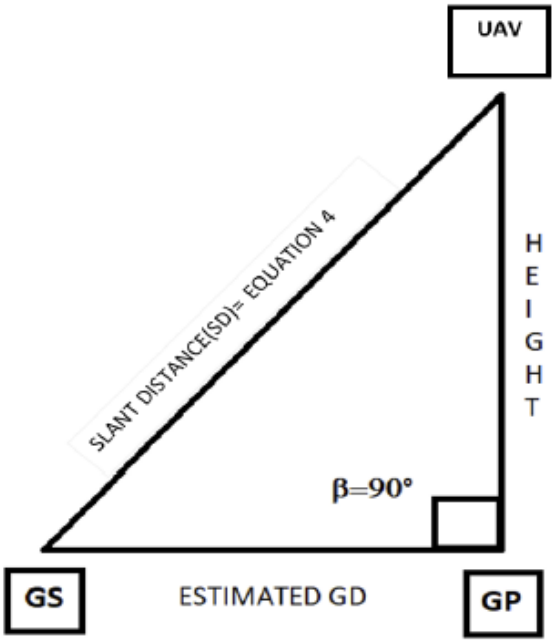

Figure 3.2: Estimation of slant distance technique.

The measurement is relatively accurate, and the slant distance can be estimated as follows:

$$SD = \sqrt{GD^2 + H^2 - (2 * GD * H * \cos(\beta))} \quad \text{(Eq. 3.6)}$$

where H is the height of the UAV, which is set to a fixed height. GD is the ground distance between the GP under the UAV and the GS, and β is the angle between the GS and the UAV. The height is fixed to take the distance as the only variable parameter to simplify the measurements. The GD and its corresponding angle are measured using the application named "MEASURE". The GP is selected to be directly below the UAV. Thus, the angle between the UAV and the GP is 90 degrees.

The angle α between the GP and the GS is measured with the laser meter. Note that, β can be calculated by subtracting α from 90 degrees [16]:

$$\beta = (90 - \alpha) \quad \text{(Eq. 3.7)}$$

Using the above technique, one can estimate the values for parameters C and L. These parameters can then be used with the measured mean RSSI value to estimate distances at other UAV positions.

In this experiment to reduce the complexities involved, we eliminate alpha (α) by placing all the ground stations on the same level. Thus, the value of beta (β) is 90°.

3.3 Architecture for Location Estimation

To estimate the location of any node we require information of two things. First, we need the location of at least three ground stations to estimate a 2-D location or we need the locations of four ground stations to estimate the 3-D coordinates and second, the distance present between the ground station and the target node must be known. The distance is calculated using any one of the previously discussed techniques for distance estimation.

In our experiment the distance is calculated based on the RSSI method. Therefore, by knowing the above two things, the location of the targeted node can be calculated. For example, in the GPS system there are approximately 31 satellites used for localization, where each satellite

broadcasts its position and time to the rest of the world. Knowing the distance of a client from a satellite, we can determine the location of the client in spherical orbit using the concept of triangulation. The general architecture for location estimation is as follows:

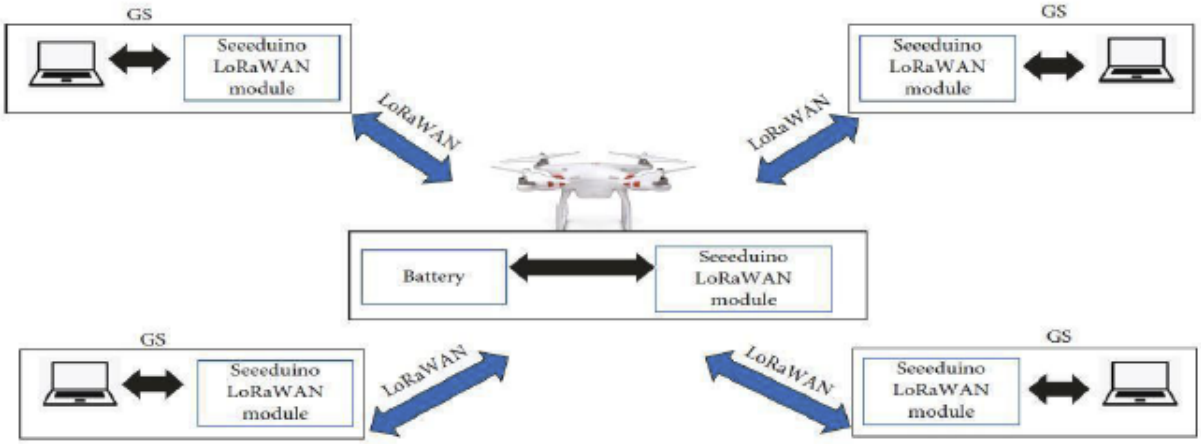

Figure 3.3: The architecture used for the location estimation [16].

To estimate the position, we calculate the slant distance (SD) between the UAV and four GSs which are used. We know that the LoRaWAN board needs a minimum of two seconds between the two consecutive messages to transmit them. To meet this criterion, the UAV must remain stationary in one fixed place for a certain amount of time. Later, the 3-D location of the UAV is determined using the trilateration technique [17] [18]. This method helps you to determine the exact position of any object in three dimensions.

CHAPTER 4

ALGORITHM IN THE MODEL

4.1 Trilateration System and Its Architecture

Trilateration is an alternative to triangulation that relies upon distance measurements and is also used to determine the 3D location with the help of Global Positioning System and receivers [19] . Using this concept, knowing about four locations in the 3-D spheres along with their distances to the destination point on an imaginary sphere we can estimate the location of the UAV. The general architecture depicting the concept of trilateration is as follows:

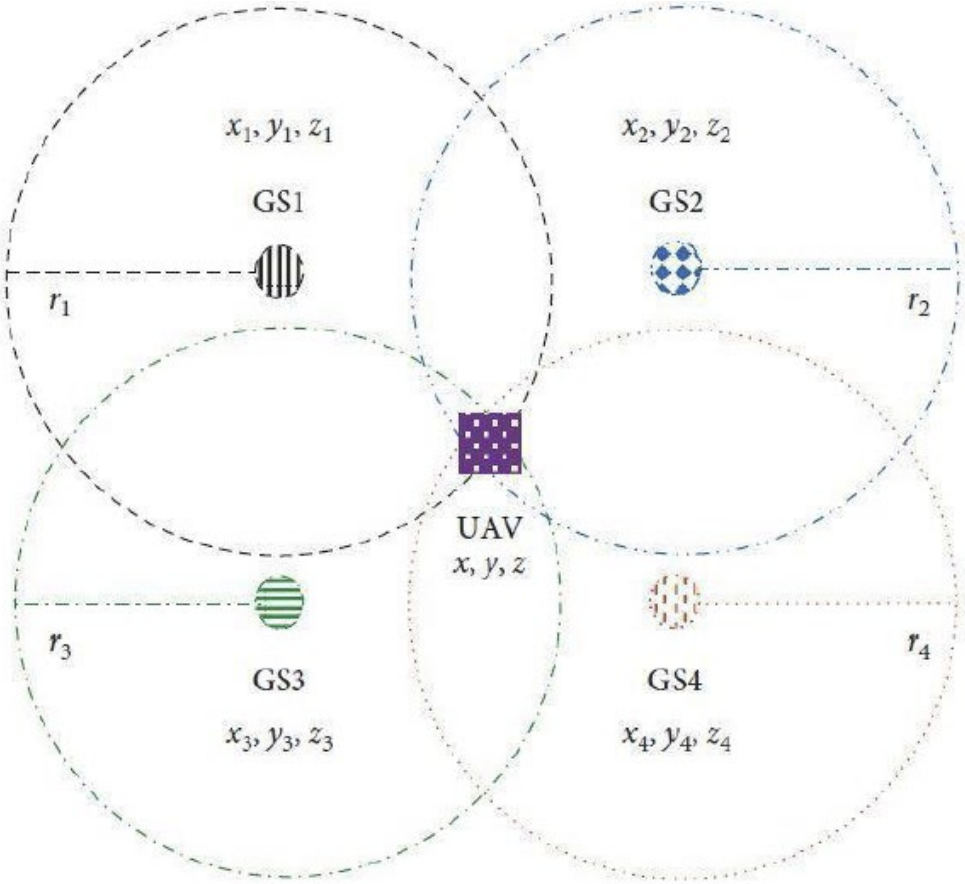

Figure 4.1: Architecture for the trilateration system [20].

Let us assume that the UAV is located at GSi on the surface of a sphere with radius (ri). For each GS, the ri equals SDi. The UAVs position is represented by a three-element vector w=x,

y, and z. The intersection of all the four spheres can be used to calculate it. All the four spheres are represented by the three-dimensional position of each GS as well as the distance between it and the UAV. Thus, using the distance formula, we can get the following four equations:

$$\begin{aligned} r_1^2 &= (x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2, \\ r_2^2 &= (x - x_2)^2 + (y - y_2)^2 + (z - z_2)^2, \\ r_3^2 &= (x - x_3)^2 + (y - y_3)^2 + (z - z_3)^2, \\ r_4^2 &= (x - x_4)^2 + (y - y_4)^2 + (z - z_4)^2. \end{aligned} \quad \text{(Eq. 4.1)}$$

where (x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3) and (x_4, y_4, z_4) are the 3-D locations of the four Ground stations. Now expanding the squares on the right-hand side of the above equations and rewriting them, we get the set of below equations:

$$\begin{aligned} r_1^2 &= x^2 - 2x_1 x + x_1^2 + y^2 - 2y_1 y + y_1^2 + z^2 - 2z_1 z + z_1^2, \\ r_2^2 &= x^2 - 2x_2 x + x_2^2 + y^2 - 2y_2 y + y_2^2 + z^2 - 2z_2 z + z_2^2, \\ r_3^2 &= x^2 - 2x_3 x + x_3^2 + y^2 - 2y_3 y + y_3^2 + z^2 - 2z_3 z + z_3^2, \\ r_4^2 &= x^2 - 2x_4 x + x_4^2 + y^2 - 2y_4 y + y_4^2 + z^2 - 2z_4 z + z_4^2. \end{aligned} \quad \text{(Eq. 4.2)}$$

To reduce them to three equations, we subtract the last equation from the first three equations. Later we arrange all the x, y, z terms on one side of the equation and all the terms of r on the other side of the equations. The resulting set of modified equations is as follows:

$$\begin{aligned} &2(x_4 - x_1)x + 2(y_4 - y_1)y + 2(z_4 - z_1)z \\ &\quad = r_1^2 - r_4^2 - x_1^2 - y_1^2 - z_1^2 + x_4^2 + y_4^2 + z_4^2, \\ &2(x_4 - x_2)x + 2(y_4 - y_2)y + 2(z_4 - z_2)z \\ &\quad = r_2^2 - r_4^2 - x_2^2 - y_2^2 - z_2^2 + x_4^2 + y_4^2 + z_4^2, \\ &2(x_4 - x_3)x + 2(y_4 - y_3)y + 2(z_4 - z_3)z \\ &\quad = r_3^2 - r_4^2 - x_3^2 - y_3^2 - z_3^2 + x_4^2 + y_4^2 + z_4^2. \end{aligned} \quad \text{(Eq. 4.3)}$$

Representing the above equations in the form of the matrices, we can simplify for the value of unknowns easily. This is represented as below:

$$\begin{bmatrix} 2(x_4-x_1) & 2(y_4-y_1) & 2(z_4-z_1) \\ 2(x_4-x_2) & 2(y_4-y_2) & 2(z_4-z_2) \\ 2(x_4-x_3) & 2(y_4-y_3) & 2(z_4-z_3) \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} r_1^2-r_4^2-x_1^2-y_1^2-z_1^2+x_4^2+y_4^2+z_4^2 \\ r_2^2-r_4^2-x_2^2-y_2^2-z_2^2+x_4^2+y_4^2+z_4^2 \\ r_3^2-r_4^2-x_3^2-y_3^2-z_3^2+x_4^2+y_4^2+z_4^2 \end{bmatrix}. \quad \text{(Eq. 4.4)}$$

$$\mathbf{A} \qquad \mathbf{w} = \qquad \mathbf{b.}$$

where A here represents the matrix of coefficients of all the Ground station locations and W is the matrix of the UAV location whose value is to be estimated and the value of 'b' is purely a constant matrix which is completely related to the distances and known locations.

Now, to find the location of UAV (w), we use the concept of least square method to solve the equations in 4.4. Using the concept of matrices, the value of w from Eq. 4.4 can be obtained as follows:

$$\mathbf{w} = (\mathbf{A}^{\mathrm{T}}\ \mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{b} \quad \text{(Eq. 4.5)}$$

The matrix W represents the 3-D location of the UAV with x as its latitude, y as its longitude and z as altitude.

4.2 Algorithm in the Model

In our proposed model as discussed, we have four GSs and a UAV at a certain height whose location is to be estimated. The GSs are placed at a certain distance from one another, and all the antennas of the Seeeduino LoRaWAN board are guided as this influences the mean RSSI values. A 3.7 V LiPo battery is used to power the LoRaWAN module that is connected to the UAV.

With the help of the Arduino IDE, the corresponding code for receiving the data is compiled and uploaded to all the 4 Seeeduino LoRaWAN modules that act as receivers at the four different ground stations. Similarly, the transmitter code is uploaded to the Seeeduino LoRaWAN

board that is fixed to the UAV. Once the data transmission begins from the transmitter, we collect the data of RSSI values up to the required amount (ten values) at all the four ground stations. With the help of all the mathematical equations discussed earlier, we developed a code in python which gives the required location of UAV with the help of all above inputs. The detailed discussion about the type of inputs needed and the practical method of implementing this algorithm is explained in the next chapter.

CHAPTER 5

EXPERIMENTAL IMPLEMENTATION AND RESULTS

5.1 Distance Estimation Using RSSI Method

For implementing the model practically, we chose an outdoor environment where we can set the four ground stations properly with adequate distance. Now the UAV along with the transmitter board which is powered by a battery is flied and kept at a height of 15m (approx.). The ground point is exactly the point which is straight below the location of the UAV (see Fig. 5.1). From this ground point, the distances to the various ground stations are measured. These obtained values are represented in Figure 5.2.

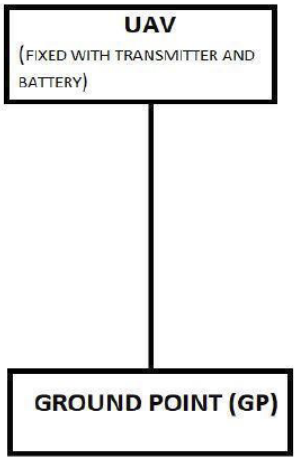

Figure 5.1: Ground point corresponding to its UAV.

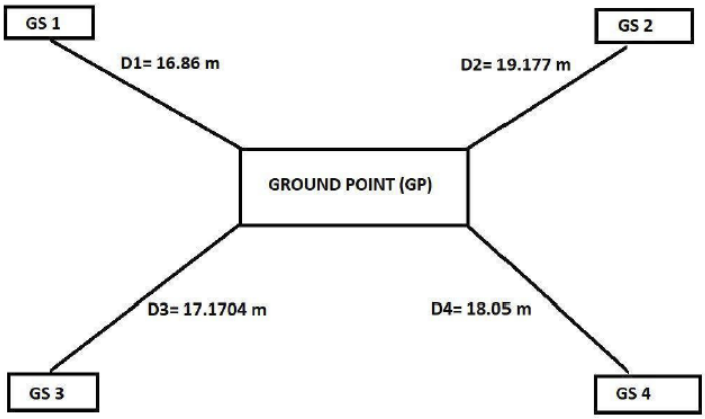

Figure 5.2: Distances of the 4 ground stations located.

All the above distances from the ground point to various ground stations were calculated using the Measurement application in the iPhone. With the help of this application, we can easily calculate the distance between two nodes. For example, for estimating the distance of GS 1 from the GP, we just need to select the start node as the ground point with the help of the screen displayed. Later we need to walk with the app open until the destined node and mark it as the end node. This automatically gives the distance which is accurate and perfect. But the obtained value is in feet and inches. This is converted to meters. All the above measured distances are in standard units of meters (m). Now, we know the data related to the various distances at which all the four ground stations are located.

The next step is to transmit the data from the transmitter that is fixed to the drone and receive the data at all the four different ground stations.

Now the corresponding code for transmitting the data is uploaded in one of the LoRaWAN modules with the help of the laptop connected to it using the Arduino IDE. After uploading the code, the transmitter is then detached from the laptop and given the power supply with the help of a 3.7 LiPo battery. Then, this transmitter is attached to the drone and flyed to a height of 15m above the level of all the four ground stations placed. Once the UAV reaches its height and stays steady, the receiving code for all the four ground stations are compiled and uploaded onto the boards with the help of the laptops connected to them. Since the transmitting is already started, once the receiving code is uploaded at all the four ground stations, the data that is being transmitted by the module that is fixed to the drone is received at all the boards present in all the four ground stations. We need to make sure that the wire antennas present on all the boards must be erected and guided properly for efficient reception of data. With the help of the serial monitor present in the Arduino IDE, the data received has been recorded at all the receivers in all the four ground

stations. The data received mainly consists of the RSSI values, length and also the data in hexadecimal format.

5.2 Location Estimation Using RSSI Method

Once we receive the set of RSSI values at all the four ground stations, we can easily estimate the slant distance of the drone using these values. The various RSSI values and their mean averages obtained across all the four ground stations are as follows:

GS1: {-60, -61, -61, -61, -66, -60, -59, -59, -60, -59} Mean Average = {-60.6}

GS2: {-74, -75, -75, -82, -79, -78, -72, -69, -68, -72} Mean Average = {-74.4}

GS3: {-70, -70, -66, -70, -68, -68, -68, -70, -69, -70} Mean Average = {-68.9}

GS4: {-74, -75, -73, -72, -74, -74, -74, -74, -74, -73} Mean Average = {-73.7}

We can see that all the values of RSSI fluctuate based on the ground distances of all the various ground stations placed. As the value of the ground distance increases, the value of RSSI also increases. This is because the propagation loss that occurs in free space increases as the distance increases. This results in weak reception of received signal value at various receivers.

As discussed previously, with the help of ground distance, height, and beta the value of slant distance is computed. And with the help of a mathematical formula for RSSI, we can also compute the values of L and C which are constants. In the algorithm, as already explained above, the concept of trilateration is used. For implementing this, we need to know the 3-D locations of all the four ground stations. These locations can easily be calculated with the help of the onboard GPS present on the Seeeduino LoRaWAN board.

For implementing it, the code for GPS must be uploaded on all the boards at all the four ground stations. Once the code is uploaded, we get the GPS data in the serial monitor at all the ground stations. Now the 3-D locations for all the four ground stations are known. With all the

equations explained in the section of algorithm, we developed a code in python that gives the output as the location of the UAV. For this developed code, the various inputs that are considered are as follows:

A total of 10 RSSI values are considered at all the four ground stations for accuracy of location estimation.

The calculated ground distance, height of the UAV (15m), beta (β) which is 90° along with all the locations of the four ground stations are given as the input for the code.

With the help of all these inputs, using the developed algorithm the 3-D location of the UAV is obtained using triangulation. This obtained location can be verified by comparing it with the on-board GPS present on the transmitter board of the Seeeduino LoRaWAN board. The code for transmitting, receiving, and finding the value of GPS for all the five Seeeduino LoRaWAN boards are in the next sections.

5.3 Seeeduino LoRaWAN as Transmitter [21]

```
#include <LoRaWan.h>

void setup(void)
{
    SerialUSB.begin(115200);
    lora.init();
    lora.initP2PMode(433, SF12, BW125, 8,
8, 20);
}

void loop(void)
{
    lora.transferPacketP2PMode("Hello
World!");
    SerialUSB.println("Send string.");
    delay (3000);
}
```

The serial USB is used to initialize the Seeeduino LoRaWAN module for the serial communication between the transmitter and receiver. There are a total of six factors that need to

be considered for the initialization of P2P mode. The value 433 indicates the tune frequency, at which the Seeeduino LoRaWAN is operated. SF12 indicates that the value of the spreading factor is 12. Spreading factor is nothing but the number of bits transmitted per second. The LoRaWAN automatically choses a factor between 7 and 12. Higher the value of the spreading factor indicates lower data rate, higher sensitivity, and longer range. The lower value of spreading factor means higher data rate, lower sensitivity, and shorter range.

BW125 is the value of bandwidth in kilohertz of the signal that has been transmitted. The value of 8 is the coding rate and the other value 8 is the length of the preamble.

5.4 Seeeduino LoRaWAN as Receiver [21]

```
#include <LoRaWan.h>

unsigned char buffer [128] = {0, };

void setup(void)
{
    SerialUSB.begin(115200);
    lora.init();
    lora.initP2PMode (433, SF12, BW125, 8, 8, 20);
}

void loop(void)
{
    short length = 0;
    short rssi = 0;

    memset (buffer, 0, 128);
    length = lora.receivePacketP2PMode(buffer, 128,  &rssi, 1);

    if(length)
    {
        SerialUSB.print("Length is: ");
        SerialUSB.println(length);
        SerialUSB.print("RSSI is: ");
        SerialUSB.println(rssi);
        SerialUSB.print("Data is: ");
        for (unsigned char i = 0; i < length; i ++)
        {
            SerialUSB.print("0x");
            SerialUSB.print(buffer[i], HEX);
            SerialUSB.print(" ");
```

```
        }
        SerialUSB.println();
    }
}
```

We receive the data in the serial monitor of all the laptops in the four ground stations and it contains RSSI values along with the message and its length.

5.5 GPS Data for the Ground Stations [21]

```
{
    Serial2.begin(9600);
    Serial.begin(115200);
}

void loop()
{
    while(Serial2.available())
    {

Serial.write(Serial2.read());
    }
    while(Serial.available())
    {

Serial2.write(Serial.read());
    }
}
void setup ()
```

Here serial.begin and serial2.begin indicate that the board utilizes two of its serial communication ports for getting the data of GPS. The values 9600 and 115200 are the baud rates. A baud rate is used to set the data rate in bits per second for the serial data transmission. For communicating with the computer, use one of these rates: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, or 115200 [22]. Once the above code is uploaded, we can find the GPS of that location by opening the serial monitor board. There are various nomenclatures of GPS that are obtained as a result in the serial monitor. These include GPGLL, GPRMC, GPVTG, GPGGA, GPGSA and GPGSV. All these nomenclatures are a part of GPS by covering

all the glossaries of GPS terminology. Among all these formats for GPS, we need a 3-D location that is represented with latitude, longitude and altitude. This can be found from the nomenclature of GPGGA where all the values are found even along with some other data. The format for GPGGA is as follows:

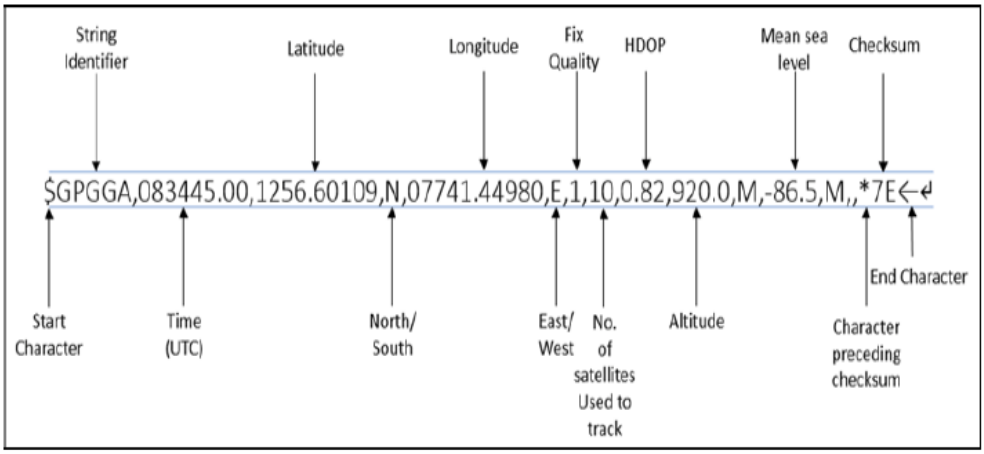

Figure 5.3: $GPGGA NMEA sentence nomenclature [23]

In the above nomenclature, UTC indicates the coordinated universal time. This is the most standard time that is commonly used across the world. All the world's timing centers maintain their synchrony by keeping UTC as their reference. Thus, the name coordinated has arrived in its abbreviation. This time is recorded at the prime meridian, where the value of longitude is 0°.

The second and the third terms indicate the latitude and longitude values of that location along with the direction. There are a total of three types of formats for representing latitudes and longitudes. These include decimal degree minutes (DDM), decimal degrees (DD) and degrees minutes seconds (DMS). All the obtained values of latitude and longitude here are in the format of degrees decimal minutes. But for our computation we change them into decimal degree format.

The GPS fix quality value generally ranges from 0 to 9 and each number has its own significance. In the above example, the value of obtained fix quality is 1 which indicates that the GPS is a single point and converging PPP of a fixed position. The various factors and their

meanings are described in the Table 5.1.

Table 5.1: GPS quality indicators [24]

Indicator	Description
0	Fix not available of invalid
1	Single point
	Converging PPP (TerraStar-L)
2	Pseudorange differential
	Converged PPP (TerraStar-L)
	Converging PPP (Terra-Star-C, TerraStar-C PRO, TerraStar-X)
4	RTK fixed ambiguity solution
5	RTK floating ambiguity solution
	Converged PPP (Terra-Star-C, TerraStar-C PRO, TerraStar-X)
6	Dead reckoning mode
7	Manual input mode (fixed position)
8	Simulator mode
9	WAAS (SBAS)

The next factor indicates the total number of satellites used to track and obtain the value of current GPS.

The next factor is HDOP which is abbreviated as horizontal dilution of precision. It tells us how accurate our obtained result is. For a better result, the value of HDOP must be smaller. Usually, it lies between one and two. If it is greater than that it indicates that the result is not that accurate.

The next factor is altitude which represents the height of the location and is usually the term ‘z’ in a 3-D location of a node. M indicates the value of the obtained altitude is in meters.

The next factor represents the mean sea level. It gives us the relation between the altitude of the antenna and the geoidal separation. Again here ‘M’ represents that the units are in meters.

The last factor indicates the checksum which tells us about the transmission errors in the received data.

5.6 Results

As already discussed in the practical implementation, there are a total of four ground stations along with UAV. All the nodes in the experiment used the LoRaWAN module. It is completely based on the Arduino zero bootloader. Hence no additional boards are required for Seeeduino LoRaWAN. The height of the UAV is 15m. As discussed earlier, in this experiment we consider a total of 10 values for the mean average of RSSI values at all the ground stations. All the results obtained here are of the serial monitor display at all the four ground stations. With the above explained setup, the values of the RSSI values obtained at all the four ground stations are as follows.

5.6.1 At GS1

```
> Length is: 12
> RSSI is: -60
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -61
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -61
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -61
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -66
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -60
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -59
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -59
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -60
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -59
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -66
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -69
> Data is: 0x44 0x44 0x44 0x44 0x44 0x44 0x44 0x44 0x44 0x44 0x44 0x44
```

Figure 5.4: Received data in serial monitor at GS1.

The ten sample values collected at GS1: {-60, -61, -61, -61, -66, -60, -59, -59, -60, -59}

Mean RSSI at GS1 = {-60.6}

5.6.2 At GS2

```
> Length is: 12
> RSSI is: -74
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -75
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -75
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -82
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -79
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -78
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -72
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -69
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -68
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -72
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -71
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -77
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -72
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -78
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -78
```

Figure 5.5: Received data in serial monitor at GS2.

The ten sample values collected at GS2: {-74, -75, -75, -82, -79, -78, -72, -69, -68, -72}

Mean RSSI at GS2 = {-74.4}

5.6.3 At GS3

```
-> Length is: 12
-> RSSI is: -70
-> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
-> Length is: 12
-> RSSI is: -70
-> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
-> Length is: 12
-> RSSI is: -66
-> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
-> Length is: 12
-> RSSI is: -70
-> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
-> Length is: 12
-> RSSI is: -68
-> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
-> Length is: 12
-> RSSI is: -68
-> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
-> Length is: 12
-> RSSI is: -68
-> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x77 0x77 0x77 0x77
-> Length is: 12
-> RSSI is: -6
-> Data is: 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
-> Length is: 12
-> RSSI is: -70
-> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
-> Length is: 12
-> RSSI is: -69
-> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
-> Length is: 12
-> RSSI is: -255
-> Data is: 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
```

Figure 5.6: Received data in serial monitor at GS3.

The ten sample values collected at GS3: {-70, -70, -66, -70, -68, -68, -68, -70, -69, -70}

Mean RSSI at GS3: {-68.9}

5.6.4 At GS4

```
> Length is: 12
> RSSI is: -74
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -75
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -73
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -72
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -74
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -74
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -74
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -74
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -74
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -73
> Data is: 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
> Length is: 12
> RSSI is: -75
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -74
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -73
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x72 0x6C 0x64 0x21
> Length is: 12
> RSSI is: -74
> Data is: 0x48 0x65 0x6C 0x6C 0x6F 0x20 0x57 0x6F 0x77 0x77 0x77 0x77
```

Figure 5.7: received data in serial monitor at GS4.

The ten sample values collected at GS4: {-74, -75, -73, -72, -74, -74, -74, -74, -74, -73}

Mean RSSI at GS4: {-73.7}

With the help of the onboard GPS available on the Seeeduino LoRaWAN board, the data of the GPS obtained at various ground stations is as follows:

5.6.5 GPS at GS1

```
-> $GPGLL,3312.7420,N,09709.5630,W,235535.000,A,D*4C
-> $GPRMC,235536.000,A,3312.7419,N,09709.5629,W,0.29,99.56,030421,,,D*46
-> $GPVTG,99.56,T,,M,0.29,N,0.53,K,D*06
-> $GPGGA,235536.000,3312.7419,N,09709.5629,W,2,5,1.46,197.6,M,-23.9,M,,*64
-> $GPGSA,A,3,06,02,12,05,19,,,,,,,,1.76,1.46,0.99*0B
-> $GPGSV,2,1,07,02,72,324,21,12,60,306,26,51,49,198,33,06,48,035,30*7E
-> $GPGSV,2,2,07,19,34,077,18,05,26,185,34,28,02,145,*4A
-> $GPGLL,3312.7419,N,09709.5629,W,235536.000,A,D*4D
-> $GPRMC,235537.000,A,3312.7418,N,09709.5629,W,0.31,99.56,030421,,,D*4F
-> $GPVTG,99.56,T,,M,0.31,N,0.58,K,D*04
-> $GPGGA,235537.000,3312.7418,N,09709.5629,W,2,5,1.45,197.8,M,-23.9,M,,*69
-> $GPGSA,A,3,06,02,12,05,19,,,,,,,,1.76,1.45,0.99*08
-> $GPGSV,2,1,08,02,72,324,20,12,60,306,26,51,49,198,33,06,48,035,30*70
-> $GPGSV,2,2,08,19,34,077,18,05,26,185,34,25,20,318,,28,02,145,*7A
-> $GPGLL,3312.7418,N,09709.5629,W,235537.000,A,D*4D
-> $GPRMC,235538.000,A,3312.7418,N,09709.5628,W,0.31,99.56,030421,,,D*41
-> $GPVTG,99.56,T,,M,0.31,N,0.58,K,D*04
-> $GPGGA,235538.000,3312.7418,N,09709.5628,W,2,5,1.46,197.9,M,-23.9,M,,*65
-> $GPGSA,A,3,06,02,12,05,19,,,,,,,,1.76,1.46,0.99*0B
-> $GPGSV,2,1,08,02,72,324,20,12,60,306,26,51,49,198,33,06,48,035,30*70
-> $GPGSV,2,2,08,19,34,077,18,05,26,185,34,25,20,318,,28,02,145,*7A
-> $GPGLL,3312.7418,N,09709.5628,W,235538.000,A,D*43
-> $GPRMC,235539.000,A,3312.7417,N,09709.5627,W,0.24,99.56,030421,,,D*44
-> $GPVTG,99.56,T,,M,0.24,N,0.45,K,D*0C
-> $GPGGA,235539.000,3312.7417,N,09709.5627,W,2,5,1.46,198.0,M,-23.9,M,,*62
-> $GPGSA,A,3,06,02,12,05,19,,,,,,,,1.76,1.46,0.99*0B
-> $GPGSV,2,1,08,02,72,324,21,12,60,306,25,51,49,198,33,06,48,035,30*72
-> $GPGSV,2,2,08,19,34,077,18,05,26,185,34,25,20,318,,28,02,145,*7A
-> $GPGLL,3312.7417,N,09709.5627,W,235539.000,A,D*42
-> $GPRMC,235540.000,A,3312.7417,N,09709.5626,W,0.24,99.56,030421,,,D*4B
-> $GPVTG,99.56,T,,M,0.24,N,0.44,K,D*0D
-> $GPGGA,235540.000,3312.7417,N,09709.5626,W,2,5,1.45,198.2,M,-23.9,M,,*6C
-> $GPGSA,A,3,06,02,12,05,19,,,,,,,,1.76,1.45,0.99*08
-> $GPGSV,2,1,08,02,72,324,21,12,60,306,25,51,49,198,33,06,48,035,30*72
-> $GPGSV,2,2,08,19,34,077,18,05,26,185,34,25,20,318,,28,02,145,*7A
-> $GPGLL,3312.7417,N,09709.5626,W,235540.000,A,D*4D
-> $GPRMC,235541.000,A,3312.7417,N,09709.5625,W,0.21,99.56,030421,,,D*4C
-> $GPVTG,99.56,T,,M,0.21,N,0.40,K,D*0C
-> $GPGGA,235541.000,3312.7417,N,09709.5625,W,2,5,1.46,198.3,M,-23.9,M,,*6C
-> $GPGSA,A,3,06,02,12,05,19,,,,,,,,1.76,1.46,0.99*0B
-> $GPGSV,2,1,08,02,72,324,22,12,60,306,25,51,49,198,34,06,48,035,29*7E
-> $GPGSV,2,2,08,19,34,077,18,05,26,185,34,25,20,318,,28,02,145,*7A
-> $GPGLL,3312.7417,N,09709.5625,W,235541.000,A,D*4F
-> $GPRMC,235542.000,A,3312.7417,N,09709.5625,W,0.19,99.56,030421,,,D*44
```

Figure 5.8: GPS data in serial monitor at GS1.

Using GPGGA the GPS at GS1 is: [3312.7417 N, 9709.5625 W, 198.3]. Here all the data is in the format of decimal degree minutes (DDM).

5.6.6 GPS at GS2

```
> $GPGSV,3,3,09,09,02,068,*45
> $GPGLL,3312.7337,N,09709.5783,W,000958.000,A,A*42
> $GPRMC,000959.000,A,3312.7337,N,09709.5783,W,0.19,228.25,040421,,,A*70
> $GPVTG,228.25,T,,M,0.19,N,0.35,K,A*3C
> $GPGGA,000959.000,3312.7337,N,09709.5783,W,1,5,1.21,218.6,M,-23.9,M,,*69
> $GPGSA,A,3,06,02,05,19,25,,,,,,,,1.53,1.21,0.93*03
> $GPGSV,3,1,09,02,70,343,21,12,65,296,,51,49,198,,06,42,039,17*78
> $GPGSV,3,2,09,05,32,185,17,19,31,084,14,25,25,316,25,17,13,096,*74
> $GPGSV,3,3,09,09,02,068,*45
> $GPGLL,3312.7337,N,09709.5783,W,000959.000,A,A*43
> $GPRMC,001000.000,A,3312.7337,N,09709.5782,W,0.29,228.25,040421,,,A*76
> $GPVTG,228.25,T,,M,0.29,N,0.54,K,A*38
> $GPGGA,001000.000,3312.7337,N,09709.5782,W,1,5,1.21,218.5,M,-23.9,M,,*6F
> $GPGSA,A,3,06,02,05,19,25,,,,,,,,1.53,1.21,0.93*03
> $GPGSV,3,1,09,02,70,343,21,12,65,296,,51,49,198,,06,42,039,17*78
> $GPGSV,3,2,09,05,32,185,17,19,31,084,12,25,25,316,25,17,13,096,*72
> $GPGSV,3,3,09,09,02,068,*45
> $GPGLL,3312.7337,N,09709.5782,W,001000.000,A,A*46
> $GPRMC,001001.000,A,3312.7338,N,09709.5782,W,0.37,228.25,040421,,,A*77
> $GPVTG,228.25,T,,M,0.37,N,0.69,K,A*39
> $GPGGA,001001.000,3312.7338,N,09709.5782,W,1,5,1.21,218.4,M,-23.9,M,,*60
> $GPGSA,A,3,06,02,05,19,25,,,,,,,,1.53,1.21,0.93*03
> $GPGSV,3,1,09,02,70,343,21,12,65,296,,51,49,198,,06,42,039,17*78
> $GPGSV,3,2,09,05,32,185,17,19,31,084,12,25,25,316,25,17,13,096,*72
> $GPGSV,3,3,09,09,02,068,*45
> $GPGLL,3312.7338,N,09709.5782,W,001001.000,A,A*48
> $GPRMC,001002.000,A,3312.7339,N,09709.5781,W,0.42,228.25,040421,,,A*74
> $GPVTG,228.25,T,,M,0.42,N,0.79,K,A*3A
> $GPGGA,001002.000,3312.7339,N,09709.5781,W,1,5,1.21,218.3,M,-23.9,M,,*66
> $GPGSA,A,3,06,02,05,19,25,,,,,,,,1.53,1.21,0.93*03
> $GPGSV,3,1,09,02,70,343,21,12,65,296,,51,49,198,,06,42,039,17*78
> $GPGSV,3,2,09,05,32,185,17,19,31,084,13,25,25,316,25,17,13,096,*73
> $GPGSV,3,3,09,09,02,068,*45
> $GPGLL,3312.7339,N,09709.5781,W,001002.000,A,A*49
> $GPRMC,001003.000,A,3312.7340,N,09709.5780,W,0.47,228.25,040421,,,A*7F
> $GPVTG,228.25,T,,M,0.47,N,0.88,K,A*31
> $GPGGA,001003.000,3312.7340,N,09709.5780,W,1,5,1.21,218.2,M,-23.9,M,,*69
> $GPGSA,A,3,06,02,05,19,25,,,,,,,,1.53,1.21,0.93*03
> $GPGSV,3,1,09,02,70,343,21,12,65,296,,51,49,198,,06,42,039,17*78
> $GPGSV,3,2,09,05,32,185,17,19,31,084,11,25,25,316,25,17,13,096,*71
> $GPGSV,3,3,09,09,02,068,*45
> $GPGLL,3312.7340,N,09709.5780,W,001003.000,A,A*47
> $GPRMC,001004.000,A,3312.7341,N,09709.5779,W,0.49,228.25,040421,,,A*71
> $GPVTG,228.25,T,,M,0.49,N,0.91,K,A*37
```

Figure 5.9: GPS data in serial monitor at GS2.

Using the GPGGA in the above obtained data, the GPS at GS2 is: [3312.7340 N, 9709.5780 W, 218.2]. Here all the values of GPS obtained are in the format of DDM.

5.6.7 GPS at GS3

```
$GPGLL,3312.7503,N,09709.5557,W,000428.000,A,D*47
$GPRMC,000429.000,A,3312.7502,N,09709.5557,W,0.32,113.81,040421,,,D*78
$GPVTG,113.81,T,,M,0.32,N,0.59,K,D*3F
$GPGGA,000429.000,3312.7502,N,09709.5557,W,2,5,1.46,192.2,M,-23.9,M,,*6F
$GPGSA,A,3,06,02,12,05,19,,,,,,,,1.77,1.46,0.99*0A
$GPGSV,2,1,08,02,71,336,27,12,63,301,18,06,44,038,28,19,32,081,17*79
$GPGSV,2,2,08,05,30,185,25,25,23,317,15,09,02,070,,40,,,*43
$GPGLL,3312.7502,N,09709.5557,W,000429.000,A,D*47
$GPRMC,000430.000,A,3312.7502,N,09709.5555,W,0.38,113.81,040421,,,D*78
$GPVTG,113.81,T,,M,0.38,N,0.71,K,D*3F
$GPGGA,000430.000,3312.7502,N,09709.5555,W,2,5,1.46,192.6,M,-23.9,M,,*61
$GPGSA,A,3,06,02,12,05,19,,,,,,,,1.77,1.46,0.99*0A
$GPGSV,2,1,08,02,71,336,27,12,63,301,18,06,44,038,28,19,32,081,18*76
$GPGSV,2,2,08,05,30,185,24,25,23,317,16,09,02,070,,44,,,*45
$GPGLL,3312.7502,N,09709.5555,W,000430.000,A,D*4D
$GPRMC,000431.000,A,3312.7501,N,09709.5556,W,0.30,113.81,040421,,,D*71
$GPVTG,113.81,T,,M,0.30,N,0.55,K,D*31
$GPGGA,000431.000,3312.7501,N,09709.5556,W,2,5,1.46,193.0,M,-23.9,M,,*67
$GPGSA,A,3,06,02,12,05,19,,,,,,,,1.77,1.46,0.99*0A
$GPGSV,2,1,08,02,71,336,27,12,63,301,18,06,44,038,28,19,32,081,18*76
$GPGSV,2,2,08,05,30,185,24,25,23,317,16,09,02,070,,44,,,*45
$GPGLL,3312.7501,N,09709.5556,W,000431.000,A,D*4C
$GPRMC,000432.000,A,3312.7500,N,09709.5559,W,0.25,113.81,040421,,,D*78
$GPVTG,113.81,T,,M,0.25,N,0.46,K,D*37
$GPGGA,000432.000,3312.7500,N,09709.5559,W,2,5,1.46,193.5,M,-23.9,M,,*6F
$GPGSA,A,3,06,02,12,05,19,,,,,,,,1.77,1.46,0.99*0A
$GPGSV,2,1,08,02,71,337,27,12,63,301,18,06,44,038,28,19,32,081,18*77
$GPGSV,2,2,08,05,30,185,25,25,23,317,15,09,02,070,,44,,,*47
$GPGLL,3312.7500,N,09709.5559,W,000432.000,A,D*41
$GPRMC,000433.000,A,3312.7498,N,09709.5558,W,0.24,113.81,040421,,,D*79
$GPVTG,113.81,T,,M,0.24,N,0.44,K,D*34
$GPGGA,000433.000,3312.7498,N,09709.5558,W,2,5,1.46,193.9,M,-23.9,M,,*63
$GPGSA,A,3,06,02,12,05,19,,,,,,,,1.77,1.46,0.99*0A
$GPGSV,2,1,08,02,71,337,26,12,63,301,18,06,44,038,28,19,32,081,17*79
$GPGSV,2,2,08,05,30,185,25,25,23,317,15,09,02,070,,44,,,*47
$GPGLL,3312.7498,N,09709.5558,W,000433.000,A,D*41
$GPRMC,000434.000,A,3312.7497,N,09709.5557,W,0.19,113.81,040421,,,D*70
$GPVTG,113.81,T,,M,0.19,N,0.34,K,D*3D
$GPGGA,000434.000,3312.7497,N,09709.5557,W,2,5,1.46,194.3,M,-23.9,M,,*69
$GPGSA,A,3,06,02,12,05,19,,,,,,,,1.77,1.46,0.99*0A
$GPGSV,2,1,08,02,71,337,26,12,63,301,17,06,44,038,28,19,32,081,16*77
$GPGSV,2,2,08,05,30,185,25,25,23,317,15,09,02,070,,44,,,*47
$GPGLL,3312.7497,N,09709.5557,W,000434.000,A,D*46
$GPRMC,000435.000,A,3312.7494,N,09709.5558,W,0.26,113.81,040421,,,D*71
```

Figure 5.10: GPS data in serial monitor at GS3.

Using GPGGA the value of GPS at GS3 is: [3312.7497 N, 9709.5557 W, 194.3]. All the data obtained here is in the format of DDM.

5.6.8 GPS at GS4

```
-> $GPGSV,3,2,11,19,30,085,,25,26,316,26,17,13,097,,09,03,067,*7C
-> $GPGSV,3,3,11,29,01,293,,49,,,,24,,,15*44
-> $GPGLL,3312.7587,N,09709.5702,W,001240.000,A,A*45
-> $GPRMC,001241.000,A,3312.7587,N,09709.5705,W,1.04,247.94,040421,,,A*7E
-> $GPVTG,247.94,T,,M,1.04,N,1.92,K,A*3E
-> $GPGGA,001241.000,3312.7587,N,09709.5705,W,1,4,3.30,207.7,M,-23.9,M,,*65
-> $GPGSA,A,3,02,12,05,25,,,,,,,,,3.43,3.30,0.96*0A
-> $GPGSV,3,1,11,02,69,346,19,12,66,293,34,06,41,040,,05,34,185,17*7C
-> $GPGSV,3,2,11,19,30,085,,25,26,316,26,17,13,097,,09,03,067,*7C
-> $GPGSV,3,3,11,29,01,293,,49,,,,24,,,14*45
-> $GPGLL,3312.7587,N,09709.5705,W,001241.000,A,A*43
-> $GPRMC,001242.000,A,3312.7574,N,09709.5746,W,2.24,256.36,040421,,,A*7F
-> $GPVTG,256.36,T,,M,2.24,N,4.16,K,A*3E
-> $GPGGA,001242.000,3312.7574,N,09709.5746,W,1,4,3.30,207.8,M,-23.9,M,,*62
-> $GPGSA,A,3,02,12,05,25,,,,,,,,,3.43,3.30,0.96*0A
-> $GPGSV,3,1,11,02,69,346,21,12,66,293,35,06,41,040,,05,34,185,14*75
-> $GPGSV,3,2,11,19,30,085,,25,26,316,27,17,13,097,,09,03,067,*7D
-> $GPGSV,3,3,11,29,01,293,,49,,,,24,,,14*45
-> $GPGLL,3312.7574,N,09709.5746,W,001242.000,A,A*4B
-> $GPRMC,001243.000,A,3312.7570,N,09709.5768,W,2.00,263.18,040421,,,A*7A
-> $GPVTG,263.18,T,,M,2.00,N,3.71,K,A*34
-> $GPGGA,001243.000,3312.7570,N,09709.5768,W,1,4,3.29,207.8,M,-23.9,M,,*63
-> $GPGSA,A,3,02,12,05,25,,,,,,,,,3.43,3.29,0.96*02
-> $GPGSV,3,1,11,02,69,346,22,12,66,293,35,06,41,040,,05,34,185,14*76
-> $GPGSV,3,2,11,19,30,085,,25,26,316,27,17,13,097,,09,03,067,*7D
-> $GPGSV,3,3,11,29,01,293,,49,,,,24,,,15*44
-> $GPGLL,3312.7570,N,09709.5768,W,001243.000,A,A*42
-> $GPRMC,001244.000,A,3312.7562,N,09709.5788,W,2.37,251.36,040421,,,A*79
-> $GPVTG,251.36,T,,M,2.37,N,4.40,K,A*38
-> $GPGGA,001244.000,3312.7562,N,09709.5788,W,1,4,3.29,207.9,M,-23.9,M,,*68
-> $GPGSA,A,3,02,12,05,25,,,,,,,,,3.43,3.29,0.96*02
-> $GPGSV,3,1,11,02,69,346,22,12,66,293,35,06,41,040,,05,34,185,14*76
-> $GPGSV,3,2,11,19,30,085,,25,26,316,28,17,13,097,,09,03,067,*72
-> $GPGSV,3,3,11,29,01,293,,33,-1,094,,24,,,15*68
-> $GPGLL,3312.7562,N,09709.5788,W,001244.000,A,A*48
-> $GPRMC,001245.000,A,3312.7554,N,09709.5803,W,2.72,241.01,040421,,,A*75
-> $GPVTG,241.01,T,,M,2.72,N,5.04,K,A*3D
-> $GPGGA,001245.000,3312.7554,N,09709.5803,W,1,4,3.29,208.0,M,-23.9,M,,*66
-> $GPGSA,A,3,02,12,05,25,,,,,,,,,3.43,3.29,0.96*02
-> $GPGSV,3,1,11,02,69,347,22,12,66,293,35,06,41,040,,05,34,185,12*71
-> $GPGSV,3,2,11,19,30,085,,25,26,316,27,17,13,097,,09,03,067,*7D
-> $GPGSV,3,3,11,29,01,293,,33,-1,094,,24,,,14*69
-> $GPGLL,3312.7554,N,09709.5803,W,001245.000,A,A*40
-> $GPRMC,001246.000,A,3312.7549,N,09709.5815,W,2.93,238.63,040421,,,A*78
```

Figure 5.11: GPS data in serial monitor at GS4.

The value of GPS is obtained from GPGGA at GS4: [3312.7554 N. 9709.5803 W, 208.0]. The format of GPS here is in DDM.

5.6.9 GPS at Transmitter

```
-> $GPVTG,257.64,T,,M,0.48,N,0.90,K,N*35
-> $GPGGA,230926.000,,,,,0,4,,,M,,M,,*40
-> $GPGSA,A,1,,,,,,,,,,,,,,*1E
-> $GPGSV,2,1,07,06,78,021,24,02,60,242,17,51,49,198,,17,34,063,26*7E
-> $GPGSV,2,2,07,28,28,130,21,14,20,134,23,03,07,038,*40
-> $GPGLL,,,,,230926.000,V,N*76
-> $GPRMC,230927.000,V,,,,,0.50,257.64,290321,,,N*4C
-> $GPVTG,257.64,T,,M,0.50,N,0.93,K,N*3F
-> $GPGGA,230927.000,,,,,0,4,,,M,,M,,*41
-> $GPGSA,A,1,,,,,,,,,,,,,,*1E
-> $GPGSV,2,1,07,06,78,021,24,02,60,242,17,51,49,198,,17,34,063,27*7F
-> $GPGSV,2,2,07,28,28,130,22,14,20,134,23,03,06,037,*4D
-> $GPGLL,,,,,230927.000,V,N*77
-> $GPRMC,230928.000,V,,,,,0.55,257.64,290321,,,N*46
-> $GPVTG,257.64,T,,M,0.55,N,1.02,K,N*33
-> $GPGGA,230928.000,,,,,0,4,,,M,,M,,*4E
-> $GPGSA,A,1,,,,,,,,,,,,,,*1E
-> $GPGSV,2,1,07,06,78,021,25,02,60,242,17,51,49,198,,17,34,063,27*7E
-> $GPGSV,2,2,07,28,28,130,22,14,20,134,23,03,06,037,*4D
-> $GPGLL,,,,,230928.000,V,N*78
-> $GPRMC,230929.000,A,3312.7499,N,09709.5634,W,0.41,257.64,290321,,,A*70
-> $GPVTG,257.64,T,,M,0.41,N,0.77,K,A*3A
-> $GPGGA,230929.000,3312.7499,N,09709.5634,W,1,4,2.44,207.7,M,-23.9,M,,*6F
-> $GPGSA,A,3,14,28,06,17,,,,,,,,,2.63,2.44,1.00*09
-> $GPGSV,2,1,07,06,78,021,26,02,60,242,17,51,49,198,,17,34,063,28*72
-> $GPGSV,2,2,07,28,28,130,22,14,20,134,23,03,06,037,*4D
-> $GPGLL,3312.7499,N,09709.5634,W,230929.000,A,A*4B
-> $GPRMC,230930.000,A,3312.7500,N,09709.5632,W,0.38,257.64,290321,,,A*71
-> $GPVTG,257.64,T,,M,0.38,N,0.70,K,A*33
-> $GPGGA,230930.000,3312.7500,N,09709.5632,W,1,4,2.44,207.9,M,-23.9,M,,*6E
-> $GPGSA,A,3,14,28,06,17,,,,,,,,,2.63,2.44,1.00*09
-> $GPGSV,2,1,07,06,77,021,26,02,60,242,17,51,49,198,,17,34,063,28*7D
-> $GPGSV,2,2,07,28,28,130,22,14,20,134,23,03,06,037,*4D
-> $GPGLL,3312.7500,N,09709.5632,W,230930.000,A,A*44
-> $GPRMC,230931.000,A,3312.7502,N,09709.5631,W,0.39,257.64,290321,,,A*70
-> $GPVTG,257.64,T,,M,0.39,N,0.72,K,A*30
-> $GPGGA,230931.000,3312.7502,N,09709.5631,W,1,4,2.44,208.0,M,-23.9,M,,*68
-> $GPGSA,A,3,14,28,06,17,,,,,,,,,2.64,2.44,1.00*0E
-> $GPGSV,2,1,07,06,77,021,26,02,60,242,17,51,49,198,,17,34,063,28*7D
-> $GPGSV,2,2,07,28,28,130,22,14,20,134,23,03,06,037,*4D
-> $GPGLL,3312.7502,N,09709.5631,W,230931.000,A,A*44
-> $GPRMC,230932.000,A,3312.7501,N,09709.5626,W,0.36,257.64,290321,,,A*79
-> $GPVTG,257.64,T,,M,0.36,N,0.66,K,A*3A
-> $GPGGA,230932.000,3312.7501,N,09709.5626,W,1,4,2.44,208.1,M,-23.9,M,,*6F
```

Figure 5.12: GPS data in serial monitor at Transmitter.

This value of GPS at the transmitter is calculated to verify the result obtained with the help of developed code in python. With the help of GPGGS the value of GPS at transmitter is as follows:

GPS at Transmitter: [3312.7500 N, 9709.5632 W, 207.9]

But for evaluating the results the data of GPS must be in the format of DMS. Hence with the help of an online spatial converter [25]. The GPS values at four ground stations in the format of DMS is as follows:

GPS at GS1: [33.212362, -97.159375]

GPS at GS2: [33.212233, -97.159633]

GPS at GS3: [33.212495, -97.159262]

GPS at GS4: [33.21259, -97.159672]

But the estimation of our 3-D location is in the format of x, y, z. Hence using the latitude, longitude, and altitude converter [26], to earth centered, earth fixed values in meters (m). the obtained data is as follows:

GPS at GS1: [-665744, -5300128, 3473796]

GPS at GS2: [-665771, -5300150, 3473795]

GPS at GS3: [-665732, -5300169, 3473806]

GPS at GS4: [-665771, -5300147, 3473823]

With the help of all the above inputs (GPS locations, RSSI VALUES, SD, H), the code developed with the help of algorithm is executed and the obtained output is as follows:

OUTPUT: [-665749.20883017, -5300149.79565052, 3473809.1388197]

Converting these x, y, z values into latitude, longitude, and altitude, we get the calculated output location of transmitter is as follows:

Calculated GPS LOCATION: [33.21235 N, -97.1594 W, 223.7]

Actual GPS LOCATION: [33.2125 N, -97.159387 W, 207.9]

Both the calculated and actual values of GPS when given in the google MAPS give the same location.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

In recent times, there has been a lot of increase in the demand of UAVs because of their various applications in various fields. Hence, a system must be developed for identifying the location of these drones flying in the air. There are a lot of solutions available in the world today that can be used for estimating the location of UAV. But most of them fail to detect the location in the non-line of sight environment. In this thesis we gave an alternative method of using the values of RSSI to estimate the drone location using the LoRaWAN modules in a GPS denied environment. The main advantage of using this system is that the LoRaWAN modules used are based on the Arduino zero bootloader and they do not require any extra module for transmitting and receiving the data. Also, these boards come with an onboard wire antenna and no external antenna is required. The cost of implementation for this project is very cheap and can easily be affordable. After setting up the arrangements for the experiment, we could successfully get the location of the drone which exactly matches the result of its actual location. There is a little variation in the altitude of the obtained UAV. Trilateration was the concept that was used for estimating the 3-D location of the UAV. This system can be used further for tracking the drone location and can also be used to locate the UAV when it enters restricted areas without permission.

6.2 Future Work

The method of using the values of RSSI for estimating the drone location seemed to be very interesting and promising. The calculated location was very close to the actual location of the UAV except a slight difference in the value of its altitude. This error might have arisen because of the improper direction of the wire antenna or maybe because of the constant parameters (L and C)

estimated for the free space environment. To overcome this issue, in future we can plan to use LoPy boards instead of using these LoRaWAN boards. With the help of these LoPy boards we can use an external antenna which helps in providing the strong signal power that can be transmitted in various directions and can support a longer distance of communication. Also, this LoPy can be used to run on ESP -32 board and as LoRaWAN it uses the physical layer. In this proposed model, all the data from the various ground stations are collected offline manually. In the future, for a faster computing system, we can connect all the systems using a gateway or switch and centralize the data to a main computer so that the computation is done automatically and produce the results very fast. In this system the UAV must stay at a fixed position for at least 10 seconds to gather the required amount of RSSI values for computation. To overcome this in future, the various wireless boards can be tested to check if there is any possibility for transmitting the data in a few microsecond intervals of time gap for consecutive messages.

REFERENCES

[1] Kang, J. H., Park, K. J., & Kim, H. (2015, October). Analysis of localization for drone-fleet. In 2015 International Conference on Information and Communication Technology Convergence (ICTC) (pp. 533-538). IEEE.

[2] "Wikipedia," semtech, [Online]. Available: https://en.wikipedia.org/wiki/LoRa.

[3] "What is LoRaWAN® Specification," LoRa Alliance, [Online]. Available: https://lora-alliance.org/about-lorawan/.

[4] "What are LoRa® and LoRaWAN®?," SemTech | LoRa, [Online]. Available: https://lora-developers.semtech.com/library/tech-papers-and-guides/lora-and-lorawan/

[5] "Seeeduino LoRaWAN," Seeed The IoT hardware enabler, [Online]. Available: https://www.seeedstudio.com/Seeeduino-LoRaWAN-p-2780.html

[6] "Seeeduino LoRaWAN," Seed the IoT hardware enabler, [Online]. Available: https://wiki.seeedstudio.com/Seeeduino_LoRAWAN/

[7] Kang, J. H., Park, K. J., & Kim, H. (2015, October). Analysis of localization for drone-fleet. In 2015 International Conference on Information and Communication Technology Convergence (ICTC) (pp. 533-538). IEEE

[8] O'Keefe, B. (2017). Finding Location with Time of Arrival and Time Difference of Arrival Techniques. ECE Senior Capstone Project. Available: https://sites.tufts.edu/eeseniordesignhandbook/files/2017/05/FireBrick_OKeefe_F1.pdf. [Accessed 2017].

[9] Liu, L., & Wang, B. (2016, November). Malware classification using gray-scale images and ensemble learning. In 2016 3rd International Conference on Systems and Informatics (ICSAI) (pp. 1018-1022). IEEE.

[10] Wang, A., Ji, X., Wu, D., Bai, X., Ding, N., Pang, J., ... & Fang, D. (2017). GuideLoc: UAV-assisted multitarget localization system for disaster rescue. Mobile Information Systems, 2017.

[11] Cheng, L., Wu, C. D., Zhang, Y. Z., & Wang, Y. (2012). An indoor localization strategy for a mini-UAV in the presence of obstacles. International Journal of Advanced Robotic Systems, 9(4), 153.

[12] Palazon, J. A., Gozalvez, J., Sepulcre, M., & Prieto, G. (2012, September). Experimental RSSI-based localization system using wireless sensor networks. In Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012) (pp. 1-4). IEEE.

[13] Hamdoun, S., Rachedi, A., & Benslimane, A. (2013, August). Comparative analysis of RSSI-based indoor localization when using multiple antennas in Wireless Sensor

Networks. In 2013 International Conference on Selected topics in mobile and wireless networking (MoWNeT) (pp. 146-151). IEEE.

[14] Tian, X., Song, Z., Jiang, B., Zhang, Y., Yu, T., & Wang, X. (2016). HiQuadLoc: An RSS fingerprinting based indoor localization system for quadrotors. IEEE Transactions on Mobile Computing, 16(9), 2545-2559.

[15] Awad, A., Frunzke, T., & Dressler, F. (2007, August). Adaptive distance estimation and localization in WSN using RSSI measures. In 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD 2007) (pp. 471-478). IEEE.

[16] Ghubaish, A., Salman, T., & Jain, R. (2019). Experiments with a LoRaWAN-Based Remote ID System for Locating Unmanned Aerial Vehicles (UAVs). Wireless Communications and Mobile Computing, 2019.

[17] Dargie, W., & Poellabauer, C. (2010). Fundamentals of wireless sensor networks: theory and practice. John Wiley & Sons.

[18] "True-range multilateration," Wikipedia, 2017. [Online]. Available: https://en.wikipedia.org/wiki/True-range_multilateration.

[19] DiBiase, D., & John, A. D. (2008). The nature of geographic information. An Open Geospatial Textbook. URL: https://www. e-education. psu. edu/natureofgeoinfo/c6 _ p12. html (hämtad 2020-04-09). Available: https://www.e-education.psu.edu/natureofgeoinfo/c5_p12.html.

[20] Ghubaish, A., Salman, T., & Jain, R. (2019). Experiments with a LoRaWAN-Based Remote ID System for Locating Unmanned Aerial Vehicles (UAVs). Wireless Communications and Mobile Computing, 2019.

[21] "Wikipedia," seeed/ The IOT hardware enabler, [Online]. Available: https://wiki.seeedstudio.com/Seeeduino_LoRAWAN/

[22] "Serial.Begin," Arduino IDE, [Online]. Available: https://www.arduino.cc/en/Serial.Begin.

[23] "MathWorks," MATLAB, [Online]. Available: https://ww2.mathworks.cn/help/supportpkg/arduino/ref/read-serial-data-from-a-gps-shield-using-arduino-hardware.html.

[24] "HEXAGON," Novatel, [Online]. Available: https://docs.novatel.com/oem7/Content/Logs/GPGGA.htm.

[25] U. O. Minnesota, "PFC Coordinate Converter," University of Minnesota, [Online]. Available: https://www.pgc.umn.edu/apps/convert/.

[26] "latitude, longitude, height // XYZ converter," [Online]. Available: https://www.oc.nps.edu/oc2902w/coord/llhxyz.htm.